グリッドコンピューティングにおける遠隔可視化に関する研究(3.2 第2回情報シナジー研究会，3．研究活動)

グリッドコンピューティングにおける遠隔可視化に関する研究

田中拓也　曽根秀昭　江原康生　小山田耕二

1　序章

1.1　本論文の背景

今日ネットワーク研究が盛んに行われている中で,「グリッドコンピューティング」という技術が注目されている.

グリッドコンピューティングとは，ネットワーク上の多数のコンピュータの資源を統合し仮想的なスーパーコンピュータとしての利用を実現し，複数のスーパーコンピュータを連携させて能力を拡大する技術である．今回は中でも，遠隔可視化技術を利用した大規模データの高速可視化についての研究を行った．スーパーコンピュータと可視化装置をネットワーク経由で繋ぎ，遠隔可視化を実現するのだが，スーパーコンピュータの計算能力に比べてボリューム表示装置 (PC) の可視化能力が不足することがありえる．その問題点が，本研究の課題といえる．

1.2　本研究の目的

本研究の目的は，以下の二点である．

1. 遠隔可視化の実現
2. 可視化処理における高速化を目指した処理待ちの削減

今回は京都大学と東北大学間において，遠隔可視化の実験を行った．

2　遠隔可視化技術

2.1　遠隔可視化技術とは

遠隔可視化技術とは，ストリーミング技術と可視化技術を統合した技術と言える．サーバ側に可視化対象データが存在し，それをクライアント側で可視化するのだが, 大規模データの場合には，単純に可視化を行うだけではクライアントの性能によっては可視化が実現できない．そこでサーバ側で大規模データを分割し，クライアント側で少しずつ可視化処理をした後に表示をすることによって，クライアントの物理メモリに収まらないような大規模データの可視化も実現できるようにする．

2.2　サーバでの処理

サーバでは，主に次のような処理を行う．

1. リソースモニタ
2. UCD ファイル読み込み
3. 部分ボクセル分割
4. UCD から Field データへの変換
5. 部分ボクセル送信

それぞれの処理について簡単に説明する．

1. クライアントの物理メモリ, 転送速度などのリソースを取得する．これによってクライアントの可視化能力の限界を指定する．
2. 可視化対象の UCD(Unstractured Cell Data; 非構造格子データ) ファイルを読み込む．
3. 可視化するボクセルデータの分割を行う．この時，分割されたボクセルデー

タの集まりを部分ボクセルと呼ぶ．これはボクセルデータをクライアントで処理できる状態にするため，リソースモニタで得た結果を基に，クライアント側で一度に処理できる範囲内に部分ボクセルの格子数を決定し，分割数を決定後，x 軸方向に等分する．

4. 部分ボクセルごとに，UCD から Field データ (ボクセルデータ) への変換を行う．
5. 部分ボクセルの送信を行い，送信が終了したら次の部分ボクセルのデータ変換を行う．

2.3 クライアントでの処理

クライアントで行われる処理は以下のようなものである．

1. データの格納
2. ボリューム・レンダリング
3. 最終画像の表示，記録

クライアントでは各ステップごとに受け取ったデータの可視化処理を行い，最後にすべての部分ボクセルの表示の重畳を行う．そうして最終的な画像を得る．

3 実験

3.1 実験の概要

実験では，京都大学をサーバ，東北大学をクライアントとし，学術研究用の super SINET を使用して実験を行った．簡単なシステム図を図 1 に示す．

今回遠隔可視化を実現するために，汎用三次元可視化ツールである AVS Express，リアルタイム可視化ツールの VisLink を使用した．

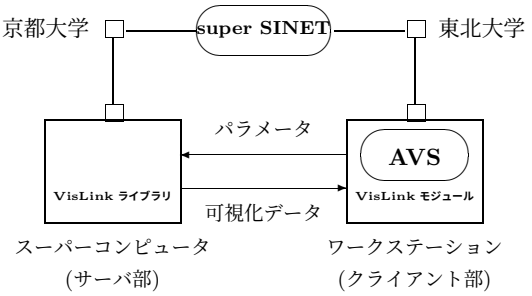

図 1　システム図

3.2 実験の手順

サーバにある可視化データを，クライアントにて可視化する．この時 Field データの各軸方向の格子数と部分ボクセル分割数を変化させ，それぞれにおけるサーバでの処理時間，通信時間，クライアントでの処理時間を記録する．

3.3 実験結果

格子数 $256 \times 256 \times 256$, $512 \times 512 \times 512$ におけるサーバ処理時間，通信時間，クライアント処理時間をそれぞれ図 2-7 に示す．ここで，通信時間はその時のネットワークの状況によって大きく変化したので，目安として受信バイト数も記録した．

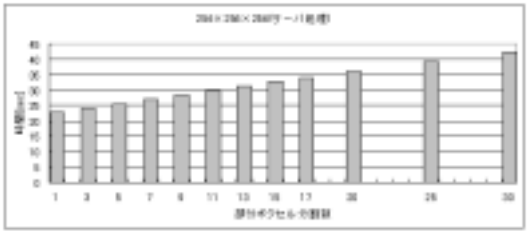

図 2　$256 \times 256 \times 256$ サーバ処理時間

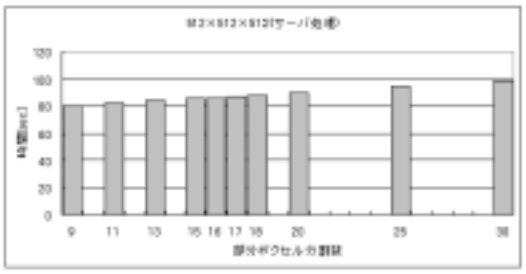

図 3　512 × 512 × 512 サーバ処理時間

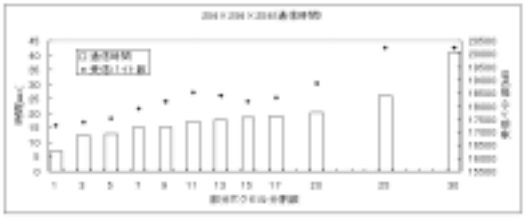

図 4　256 × 256 × 256 通信時間

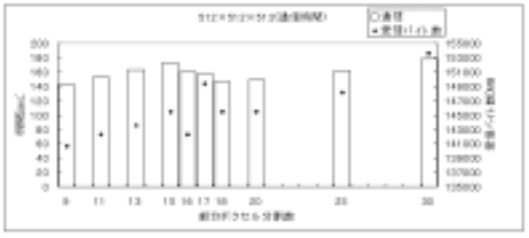

図 5　512 × 512 × 512 通信時間

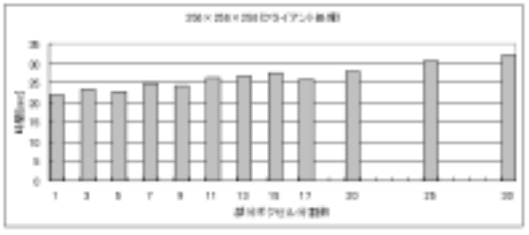

図 6　256 × 256 × 256 クライアント処理時間

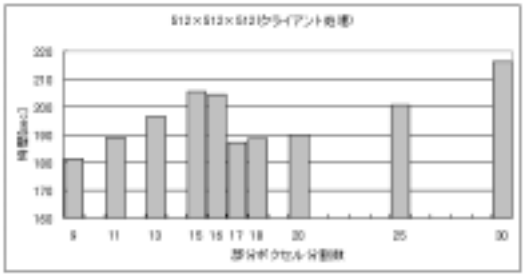

図 7　512 × 512 × 512 クライアント処理時間

4　考察

4.1　部分ボクセル分割数に関する考察

部分ボクセル分割数を増加させると，サーバでの処理時間は単調に増加した (図 2，3)．これは部分ボクセル分割の処理量が増加するためと考えられる．

対して通信時間，受信バイト数，クライアントでの処理時間は単調に増加しているとは言えない (図 4-7)．これは部分ボクセルの分割法に以下に述べる問題があるのではないかと考えられる．

ここで，格子数 256 × 256 × 256，部分ボクセル分割数 9 におけるクライアント側の各ステップ処理時間を図 8 に示す．

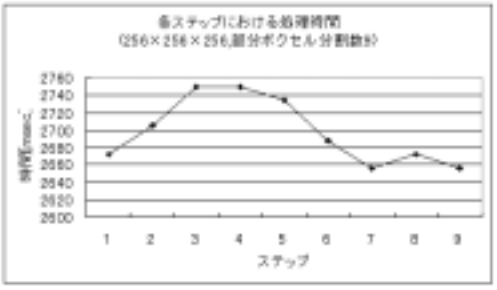

図 8　ステップごとのクライアント処理時間

図 8 より，ステップごとに処理時間に差があり，時間的に均等に分割が為されていないことがわかる．これは分割の際に単純に一方向のみに均等分割されるので、画像によって各ステップごとに処理量に差が生じてしまう．そのため，分割数の増加が必ずしも処理時間の増加を伴うわけではなくなると考えられる．このことから，一方向への均等な分割とは限らない，処理量を均等とした最適な分割方法を求めることが課題と言える．

4.2 格子数に関する考察

格子数 256 × 256 × 256，部分ボクセル分割数 9 における各時間を図 9 に示す．図 9 より，サーバでの処理時間の方がクライアントでの処理時間よりも長いことがわかる．これは各ステップにおいても同様で，サーバでの処理時間が長いため，クライアントにおいて待ち時間が存在してしまう．即ち，サーバがボトルネックになってしまっている．

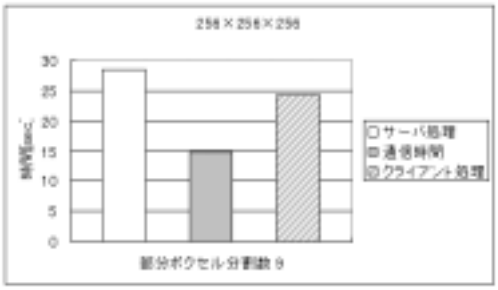

図 9　256 × 256 × 256 における各時間

次に，格子数 512 × 512 × 512，部分ボクセル分割数 9 における各時間を図 10 に示す．図 10 より，クライアントでの処理時間の方がサーバでの処理時間よりも長いことがわかる．これは各ステップにおいても同様で，今度はクライアントがボトルネックとなっている．また，格子数 768 × 512 × 512 , 768 × 768 × 768, 1024 × 1024 × 1024 でも同様の結果となった．

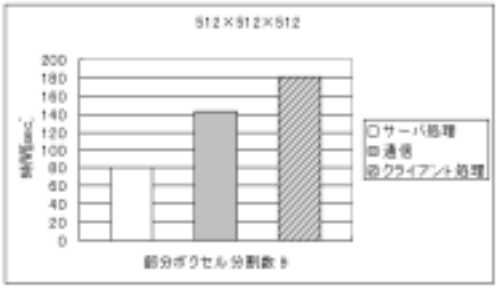

図 10　512 × 512 × 512 における各時間

以上のことより，全体の総合的性能を最大にするにはサーバ，クライアントのどちらも待ちがないことが望ましく，そのような分割方法を得る事が課題と言える．

5 結論

図 2，3 より，部分ボクセル分割数が増加すると，概ね処理時間は減少し，高速に処理を行えた．しかし，単純な分割方法ゆえ各ステップにおける処理量に差が生じてしまい，クライアントでは分割数が増加しても全体での総合的な処理時間が増加するとは限らなかった．

また，部分ボクセルに分割せずに可視化できるような小規模データではサーバ側の処理がボトルネックになり，大規模データにおいてはクライアントがボトルネックとなった．

今後の課題としては，高速可視化実現のために，以下のような事柄が挙げられる．

- 様々なクライアントのリソースに適した，汎用性のある部分ボクセル分割数の算出方法
- 可視化対象によらない, 各ステップにおいて均一な処理を行うための分割方法
- ボトルネックが生じないよう，サーバとクライアントをフルに使用するような分割方法

参考文献

[1] 平野彰雄, “京都大学におけるグリッド研究の現状と今後の計画について,” http://www.kudpc.kyoto-u.ac.jp/Archives/PDF/NewsLetter/2003-3_Grid.pdf,(2003 年)

[2] 高橋一郎, “リアルタイム可視化ツール VisLink の紹介,” http://sora.nagoya-u.ac.jp/visplus/Manual.VisLink.PDF

[3] 中村真二 “ 大規模データの高速可視化サブシステム ”